Hardware-in-the-Loop for Active Flow Control in Aerodynamics

Part Time PhD Student: Jaime Bowen Varela
Supervisor/s: Rodrigo Castellanos and Francisco Barranco(Universidad de Granada)

PhD Aerospace Engineering UC3M
Doctoral Meetings 2026
2nd and 3rd June 2026

Outline

10-Minute Talk Map

- **Motivation / problem:** Why active flow control needs hardware-aware validation.
- **State of the art critique:** Where simulation-only, control-only, and learning-only validation fall short.
- **Research goals:** Build a progressive path from benchmark dynamics to FPGA-oriented HIL.
- **Methods and experiments:** Landau oscillator, LQR baseline, residual RL, delay and constraint sweep.
- **Results and discussion:** Preliminary results show average cost improvement over LQR under realistic constraints.
- **Future and achievements:** Drone transition, hardware deployment path, and academic outputs so far.

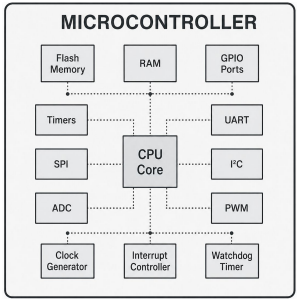
Motivation

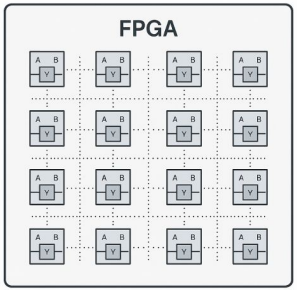
Where control ideas often live

- Simulation-first validation.
- Clean models and ideal timing.
- Limited exposure to hardware constraints.

Where aerospace systems run

- Embedded digital controllers.
- Strict timing and latency budgets.
- Sensor, actuator, and implementation limits.





Goal: Connect aerodynamic control research with realistic hardware execution constraints.

Problem and Research gap

Traditional Validation Gaps

- High-fidelity CFD is expensive and slow for real-time iterations.
- Wind-tunnel experiments are valuable but hard to iterate.
- Most control ideas are validated mainly as algorithms in simulation.

Active Flow Control Realities

- Nonlinear, fast, and highly uncertain dynamics.
- Extremely sensitive to actuation and sensing delays.
- Performance gap between software models and real hardware.

The Research Gap

The missing piece is a validation path that tests controllers as complete real-time systems, preserving realistic timing and hardware constraints.

State of the Art Critique and Research Gaps

Classical control	Strong and hardware-friendly, but often designed around linearized or idealized models.
Learning control	Can compensate nonlinear effects, but may be large, opaque, or hard to validate.
CFD / wind tunnel	Physically meaningful, but expensive and slow for controller iteration.
HIL validation	Mature in embedded domains, but rarely connected early to active-flow-control algorithm design.

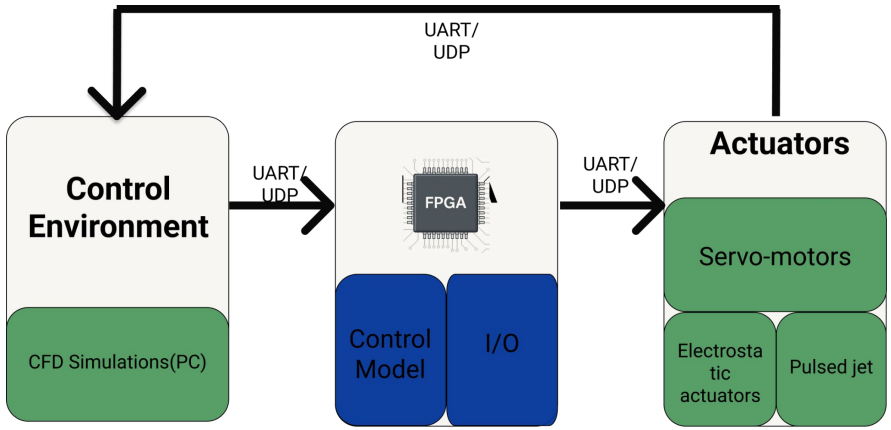
The Research Gap

The missing piece is a validation path that tests controllers as complete real-time systems.

Research Questions and goals

Research Questions

- **RQ1:** Can a small residual learning policy improve a strong LQR baseline on a nonlinear benchmark?
- **RQ2:** Under which realistic constraints does it help: delay, saturation, slew rate, noise, or model mismatch?
- **RQ3:** Can the learned controller stay small enough for fixed-point, FPGA-oriented implementation?



Approach

Progressive validation path: From nonlinear benchmarks to FPGA-oriented real-time validation and drone simulation.

Primary Goal

Build a repeatable HIL methodology before moving to higher-fidelity aerodynamic and drone simulations.

Method: From Simulation to Real-Time Control Hardware

1. **Nonlinear Benchmark**

   Landau oscillator

2. **Strong classical baseline**

   LQR.

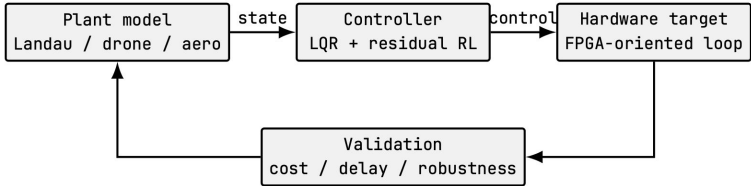
3. **Small learned correction:**

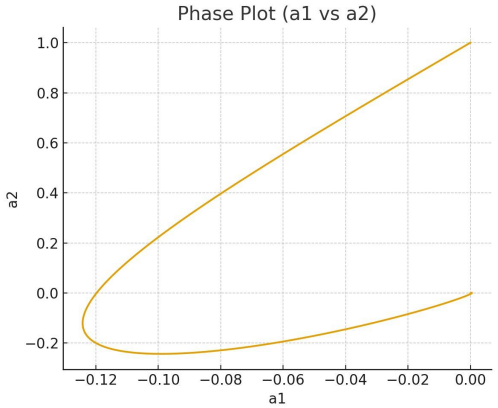
   Residual RL, 41 parameters

4. **Hardware-aware constraints:**

   Delay, saturation, slew rate, noise, mismatch

5. **HIL Execution**
   FPGA-oriented loop and drone simulation





A progressive validation path before expensive CFD, wind-tunnel, or flight testing.

Method: Benchmark and Controller

Landau Oscillator

$$\dot{a}_1 = (\mu - \beta r^2)a_1 - \omega a_2$$

$$\dot{a}_2 = (\mu - \beta r^2)a_2 + \omega a_1 + u$$

$$r^2 = a_1^2 + a_2^2$$

Used as a controlled nonlinear benchmark for active flow control strategies.

Residual Control

$$u = u_{\text{LQR}} + u_{\text{res}}$$

$$u_{\text{res}} = \alpha \tanh(W_2 \tanh(W_1 \phi + b_1) + b_2)$$

$$\phi = [a_1,\ a_2,\ u_{\text{prev}}/u_{\text{max}}]$$

Combines a strong classical LQR baseline with a learning-based correction.

Controller Size

- Small residual Neural Network: **3** → **8** → **1** architecture.
- Only **41 trainable parameters**, making it a credible candidate for FPGA deployment.

The goal is to test if a **small residual policy** can improve LQR under **realistic hardware constraints**.

Experiments and Simulations

Training loop

- Random initial conditions.
- Differentiable closed-loop rollout.
- Adam optimization.
- Best policy retained per delay value.

$$J = \Sigma\, \Delta t\, (r_k^2 + 0.05\omega_k^2 + 5p_k^2) + w_f\, r_N^2$$

Evaluation conditions

- Monte Carlo simulations.
- Delay sweep: 0, 2, 4, ..., 20 steps.
- Saturation and slew-rate constraints.
- Process and measurement noise.
- Matched and mismatched model tests.

$$J_{eval} = \Sigma\, \Delta t\, (r_k^2 + 0.05\omega_k^2)$$

Cost Function: The control objective J penalizes deviation from the origin (r), control effort (ω), and pressure gradient (p) for the training phase.

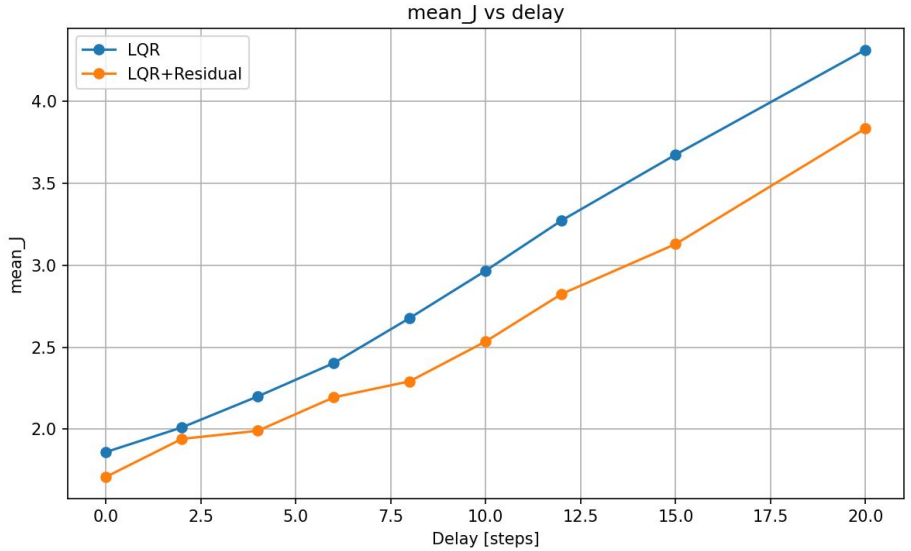
Operational Focus: Testing is performed across a delay sweep to identify the performance limits of the residual learner compared to LQR.

The setup enables identifying **peak gain regimes** and **stability boundaries** under hardware-aware constraints.

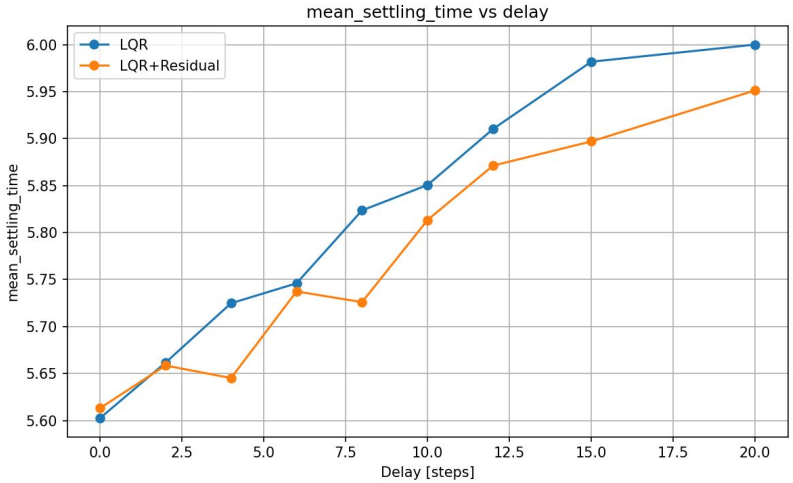
Preliminary Results: Cost

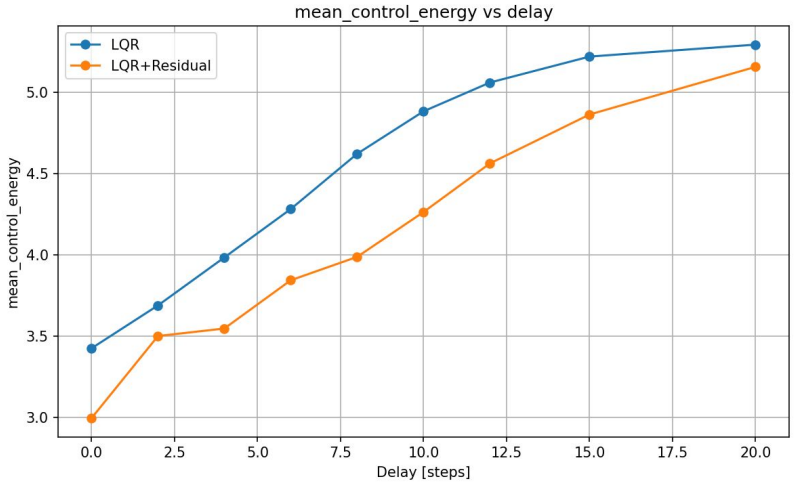
+9.3% average improvement over nominal LQR

Peak gain is approximately 13% in the mid-delay regime. No instability was observed in the tested cases.



Preliminary Results: Secondary Metrics





Cost, control energy, and settling time are all needed to understand whether residual learning improves trajectory quality or simply spends more control.

Discussion

What the results suggest

- LQR is already strong in simple cases.
- Residual learning helps most when delay and constraints make the nominal loop less ideal.
- A 41-parameter correction is a credible candidate for hardware-oriented implementation.

Remaining questions

- Which constraint regimes produce robust improvement?
- How sensitive is the result to training and initialization?
- What fixed-point precision is enough for FPGA deployment?

The results indicate that **residual reinforcement learning** effectively complements classical control in **hardware-constrained environments**.

Conclusions and Future Directions

Active flow control needs controllers that are not only good in simulation, but also fast, deterministic, and realistic enough to run on hardware.

Short-term

Strengthen Monte Carlo evaluation, mismatch cases, delay scenarios, and reproducible experiment tracking.

Medium-term

Move the residual policy toward fixed-point implementation and measure latency, determinism, and resource cost.

Next platform

Use Gazebo, ArduPilot, and possibly ROS as an operational drone simulation step before higher-fidelity active-flow-control models.

ARDUPILOT

GAZEBO

Thank you!
Questions?

Hardware-in-the-Loop for Active Flow Control in Aerodynamics

Part time PhD Student: Jaime Bowen Varela
Supervisor/s: Rodrigo Castellanos and Francisco Barranco (Universidad de Granada)

Want to connect?